UNCLASSIFIED

DEFENCE RESEARCH ESTABLISHMENT
CENTRE DE RECHERCHES POUR LA DÉFENSE
VALCARTIER

DREV - TR-2000-157
Unlimited Distribution/Distribution illimitée

AN ADAPTIVE INTELLIGENT SYSTEM ARCHITECTURE FOR TACTICAL MISSION PLANNING

by
J. Berger

November/novembre 2001

Approved by/approuvé par

Section Head/Chef de section

17 octobre 2001
Date

SANS CLASSIFICATION

WARNING NOTICE

The information contained herein is proprietary to Her Majesty and is provided to the recipient on the understanding that it will be used for information and evaluation purposes only. Any commercial use, including use for manufacture, is prohibited. Release to third parties of this publication or of information contained herein is prohibited without the prior written consent of DND Canada.

© Her Majesty the Queen in Right of Canada as represented by the Minister of National Defence, 2001

P516658.PDF [Page: 6 of 44]

ABSTRACT

Current tactical mission planning systems operating in a highly dynamic and uncertain environment can work in real-time but cannot guarantee efficient solutions. Potential constraints and conditions associated with mission criticality impose the development of adaptive reasoning capabilities to support resource planning, monitoring and control. A blackboard-based approach for an adaptive intelligent system targeted to air vehicle tactical mission planning is presented. The proposed approach consists in evolving the opportunistic nature of the basic blackboard paradigm to support real-time dynamic planning and execution control in a time-varying and uncertain environment. The parallel blackboard architecture allows for the concurrent execution of multiple knowledge sources and provides explicit mechanisms to support event-driven and goal-directed problem-solving. It also includes resource-bounded reasoning capabilities represented through a meta-level control of computation responsible for deliberation scheduling and run-time monitoring of anytime knowledge sources. Based upon an object-oriented approach, a prototype of the baseline architecture has been partly implemented. This initiative is a significant contribution toward the development of an automated real-time advisory decision support system for resource management.

RÉSUMÉ

Les systèmes de planification tactique actuels qui évoluent dans un environnement dynamique incertain peuvent fonctionner en temps réel mais ne peuvent garantir de solutions efficaces. Les contraintes et conditions potentielles associées à un contexte de mission critique imposent le développement de capacités de raisonnement adaptatifs afin d'appuyer la planification des ressources, le monitorage et le contrôle. Un système intelligent adaptatif basé sur une approche par tableau noir pour un problème de planification de mission de véhicules est présenté. L'approche consiste à étendre la nature opportuniste du paradigme tableau noir classique pour appuyer la planification dynamique et le contrôle de l'exécution dans un environnement changeant et incertain. L'architecture de type tableau noir parallèle proposée, permet l'exécution concurrente de sources de connaissances multiples et, présente des mécanismes explicites appuyant la résolution de problèmes sur la base des événements et des buts. L'architecture inclut des capacités de raisonnement adaptatifs représentées via un méta-niveau de contrôle de calcul gouvernant la délibération et le monitorage de l'exécution de sources de connaissances interruptibles. Utilisant une approche orientée objet, un premier prototype

P516658.PDF [Page: 7 of 44]

a été partiellement implanté. Cette initiative est une contribution significative dans la mise au point d'un système conseil en temps réel pour la gestion des ressources.

TABLE OF CONTENTS

ABSTRACT/ RÉSUMÉ I

EXECUTIVE SUMMARY V

1.0 INTRODUCTION 1

2.0 BLACKBOARD APPROACH 3

3.0 AIS ARCHITECTURE OVERVIEW 6

3.1 BLACKBOARD DATA STORAGE 7

3.2 KNOWLEDGE SOURCES 9

3.3 CONTROL UNIT 10

3.4 COMMUNICATION 18

4.0 APPLICATION DOMAIN 21

5.0 RELATED WORK 22

6.0 CONCLUSION 23

7.0 ACKNOWLEDGMENTS 25

8.0 REFERENCES 26

FIGURES 1 TO 9

INTENTIONNALY BLANK

EXECUTIVE SUMMARY

Operational and tactical mission planning required to support domestic, emergency management or contingency operations, or to maintain the sovereignty of Canadian airspace, calls for outstanding closed-loop command and control system flexibility to direct, plan and monitor operations and properly achieve air defence resource management. The need to provide automated capabilities to support dynamic mission planning and execution control in a time-constrained and uncertain environment is a primary and very challenging issue for air command. This is motivated by the very nature of a time-varying environment, the hierarchical structure (organization, coordination and execution levels) imposed to carry out abstract or detailed mission planning, intrinsic system boundaries characterized by limited adaptiveness and limited information processing capabilities to ensure timely resource utilization, coordination and solution quality.

Current reactive mission planning systems operating in a highly dynamic and uncertain environment can work in real-time but cannot guarantee efficient solutions, whereas deliberative systems are able to provide optimal solutions conditionally to the perfect knowledge of the world and the use of unbounded computational resources. But a real computer/human planning agent does not have a complete knowledge of the domain. It must address the problems of uncertainty, incomplete information domain, finite and limited computational power. Potential constraints and conditions associated with mission criticality impose the development of flexible or adaptive reasoning capabilities to support resource planning, monitoring and execution control.

An adaptive intelligent system architecture to support flexible resource management in a time-constrained uncertain environment is proposed. An adaptive intelligent system is a system designed to modify its behavior in response to a dynamic and uncertain environment to successfully reach its goals. The proposed blackboard-based approach is targeted to air vehicle tactical mission planning (vehicle routing and aircrew scheduling). The approach consists in evolving the opportunistic nature of the basic blackboard paradigm to support real-time dynamic planning and execution control in a time-varying uncertain environment. Well suited to handle progressive reasoning, the parallel blackboard architecture allows for the concurrent execution of multiple knowledge sources and provides explicit mechanisms to support event-driven and goal-directed problem-solving. It also includes resource-bounded reasoning capabilities represented through a meta-level control of computation responsible for deliberation scheduling and run-time monitoring of anytime knowledge sources. Using an object-oriented approach, a prototype of the baseline architecture has been partly implemented.

This initiative is a significant contribution toward the development of an automated real-time advisory decision support system for resource management. The development of an adaptive intelligent system architecture and supporting capabilities to plan, direct and control operational and tactical assets is a significant step forward in addressing key system requirements such as speed, timeliness, responsiveness and graceful adaptation. Targeted application domains involving adaptive mission planning include Army Aviation Fleet, Fighter Fleet, Air Transport Fleet and Maritime Air Fleet activities, at Wing and Squadron levels (transport, surveillance, identification, interception, intervention or destruction, contingency operations, tactical airlift, emergency management, air traffic control and other air mission management activities).

1.0 INTRODUCTION

A suitable supporting capability to dynamically plan, direct and control tactical assets in near real-time is key to various air force organizations to perform critical tasks such as tactical mission planning. An air vehicle tactical mission is composed of a series of tasks with precedence constraints. Such a mission may consist, for instance, in picking up and/or delivering resources to various locations using limited renewable air defence/transport resources (e.g. air vehicles and crew) subject to various constraints. Typical constraints include maintenance operations, mission requirements (capacity, configuration, crew), precedence relationships and soft/hard deadlines. A sequence of unexpected events may have major consequences on the feasibility and suitability of actions already planned and being executed, as the evolving situation imposes plan revision (partial or complete replanning/task rescheduling). These contingencies make the performance capability of air command and control mission planning systems very sensitive to the severity of environmental and operational conditions. Consequently, given the criticality, resource requirements and time characteristics of the current missions as well as the complexity and uncertainty of the environment (e.g. weather conditions, resource consumption, aircraft failures or time pressure), significant efforts directed to the development of an adaptive intelligent system (AIS) are required in order to timely support resource planning, monitoring and control. An adaptive intelligent system is a system designed to modify its behavior in response to a dynamic and uncertain environment in order to successfully reach its goals.

This document focuses on technology concepts proposed, developed and partly implemented for a baseline adaptive intelligent system architecture for resource management targeted to air vehicle tactical mission planning. The proposed approach is part of an effort aimed at developing advanced technology concepts and tools for an adaptive intelligent system to assist the military community in the management of air defence resources related to air command and control activities. The chosen approach

borrows from the basic "blackboard" paradigm [1]. The underlying blackboard concept relies on the opportunistic interaction of multiple experts (knowledge sources) sharing and communicating information through a common blackboard. Specialists cooperatively contribute to the overall problem-solving process (solution of a common complex problem) by gradually solving a hierarchy of related subproblems. The information is shared through a common workspace or shared memory called blackboard. The timely interaction and execution of suitable sources of expertise is mediated by a special control unit. The proposed approach consists in evolving the basic blackboard paradigm, addressing its intrinsic deficiencies and suitably adapting the problem-solving process to support real-time planning in time-varying and uncertain environment. The proposed parallel blackboard architecture allows for the concurrent execution of multiple knowledge sources (KSs), while enhancing the control unit by adding explicitly goal-directed control and a meta-level control mechanism to support resource-bounded reasoning. The meta-level control of reasoning is responsible for deliberation scheduling and run-time monitoring of anytime algorithms [2], and then explicitly introduces a mechanism to interleave dynamic plan construction and plan execution. A basic prototype exploiting various technologies and advanced concepts developed for the baseline architecture of an adaptive intelligent system has been partly implemented. The basic kernel implementation of the parallel blackboard mainly involves four major components, namely the blackboard data storage, the knowledge sources, the control unit and the communication manager. It focuses upon a centralized control process which supervises the execution of multiple knowledge sources running concurrently and interacting through the blackboard data storage.

The report is outlined as follows. Chapter 2 presents some of the critical issues to be considered as well as the main characteristics of the blackboard solution approach for the proposed adaptive intelligent system. Then, Chapter 3 gives an overview of the baseline system architecture. It namely depicts the high-level features of the system components. A brief description of the targeted application is presented in Chapter 4 and,

a short discussion about related work on blackboard is succinctly given in Chapter 5. Finally, a conclusion as well as a brief description of future work and research directions are outlined in Chapter 6.

This work was conducted at DREV in conjunction with APG Inc. and University of Alberta between January 1997 and December 1999 within Work Unit 13dm11: *Adaptive Intelligent System* of project 3dm: *Mission Planning*.

2.0 BLACKBOARD APPROACH

Real-time intelligent control for AIS is the subject of extensive research as new challenges emerge [3], [4]. System architectures developed so far generally revolves around the combination (embedded component) or the cooperation of real-time components with artificial intelligence systems [3], [4]. A popular approach relies on the use and refinement of the generic principles of the blackboard paradigm [1]. The blackboard can be seen as a communication medium providing a representation for the problem-solving process. Building blocks include knowledge sources comprising specialized problem-solving expertise, a control loop to schedule knowledge source execution and a data storage (blackboard) to hold and communicate information. A formal description of the blackboard model and architecture may be found in [5-8]. A concurrent blackboard architecture is being explored as a baseline for an adaptive intelligent system to investigate technology concepts related to concurrent dynamic planning and execution control. The proposed parallel blackboard architecture is believed to be a suitable framework to handle task complexity, multiple interactions among concurrently executing agents, communicating agents with heterogeneous sources of information and resource-bounded reasoning issues. A parallel blackboard system mainly refers to a blackboard model supporting the concurrent execution of knowledge sources [9].

P516658.PDF [Page: 15 of 44]

Many deficiencies related to the basic blackboard paradigm have to be further addressed to satisfactorily define an adaptive intelligent system. One of the key ideas of the blackboard model is that problem-solving should be both incremental and opportunistic [7]. These limitations are partly associated with some of the general themes stated by Lesser [7] to describe the evolution of blackboard control: formalization and extensions of goal-directed control without loss of opportunism, development of models to support reasoning about actions, the specification and application of explicit context-specific control strategies and the cost of control reasoning. Typical deficiencies [10] also include the atomicity of knowledge sources which prevent interruptibility while executing, sequential KS execution, the lack of goal structure to capture goal-directed problem-solving, insufficient information for planning and scheduling and the overlook of resource usage by knowledge sources making resource-bounded reasoning very difficult. These drawbacks just added to the complexity of current issues already associated to the design of concurrent blackboard architectures [9]. As a matter of fact, until recently, technology limitations impeded or at least made very laborious significant progress upon concurrent blackboard development. The primitive nature of current parallel hardware and the lack of sophisticated software development environments then available prevented real implementations. Simulated models of parallel environments (simulated parallelism) to represent concurrently executing processes and inter-process communication were believed to be a convenient alternative to overcome this problem. As a result, few complete system designs went beyond the simulation or feasibility study stage. On the other hand, an alternate formal model for blackboard systems, as well as supporting tools and methodology to design and test concurrent blackboard system architectures were recently proposed by McManus and Bynum [11]. Even though the proposed design and analysis techniques and related tools have some value in developing generic concurrent blackboard, the methodology may prove difficult to apply to all classes of problems as the basic assumptions of the approach do not necessarily hold. Indeed, the assumption about alternate ways to efficiently deal with tightly coupled

dependent knowledge sources and/or the existence of multiple domain KS decomposition schemes is believed to seriously limit the applicability of the approach in some cases.

The proposed baseline architecture for AIS introduces new features to the basic blackboard framework providing more flexibility and adaptiveness. Accordingly, the control loop extends opportunistic problem-solving (event-driven control) by adding explicit goal structure representation and mechanisms to manipulate these goals, a meta-level control of planning (deliberation scheduling and a run-time monitoring mechanism) based on anytime computation (interruptible knowledge source) to support reasoning (resource-bounded) about actions. Besides, the blackboard data storage system addresses concurrency control issues in supporting concurrent data access by knowledge sources. At the architecture level, AIS takes advantage of anytime computing [2] to make knowledge sources interruptible, while running concurrently. Exploiting multiprocessing technology as well as the use of multithreading capabilities, parallelism is primarily introduced to ensure adaptiveness to the AIS baseline architecture even though speed-up still remains a major concern. As for external communication with its surrounding environment (agents, blackboard, external information sources), the blackboard-based AIS comprises a communication manager, a key component of the baseline architecture in performing information transactions.

A basic prototype exploiting various technologies and advanced concepts developed for the baseline architecture of an adaptive intelligent system have been partly implemented. The basic kernel implementation of the parallel blackboard mainly involves four major components, namely the blackboard data storage, the knowledge sources, the control unit and the communication manager. It focuses upon a centralized control process which supervises the execution of multiple knowledge sources running concurrently and interacting through the blackboard data storage. Implemented as a shared memory, the data storage contains a collection of objects (domain and control) and data structures available to other blackboard components. Each knowledge source seen as a producer/consumer can access concurrently and independently working memory

(blackboard data storage) and achieve transactions on information using a dedicated query language. Built as a primitive library, the proposed query language has been designed to address critical concurrency issues. As a result, knowledge sources can trigger on new information deposited on the blackboard (data storage) and further be activated by the control unit. Concurrently running knowledge sources have been implemented through multithreading in a multiprocessing environment whose activation and execution are closely managed by the control component. As a separate thread of control, the control unit implementation incorporates mechanisms to ensure data/goal-directed control and support resource-bounded reasoning. Finally, the communication manager component represented as an additional thread, integrates KQML (Knowledge Query Manipulation Language) and KIF (Knowledge Interchange Format) to achieve interaction with the external environment. The object-oriented blackboard kernel has been implemented on a multiprocessing Sun platform environment using C++ and was successfully interfaced with a relational database.

3.0 AIS ARCHITECTURE OVERVIEW

The architecture of the proposed adaptive intelligent system is primarily based upon a specific interpretation of the general blackboard paradigm [1]. However, part of the concurrent parallel blackboard architecture for AIS has been inspired from some of the principles and key features described and available in BB1 [12]. An overview of the global blackboard-based architecture is given in Figure 1. The major components include the blackboard data storage, a collection of domain and control knowledge sources, the control unit to manage KSs activation/execution and, the communication manager to ensure information exchange with the environment and communication with external agents via suitable knowledge sources.

P516658.PDF [Page: 18 of 44]

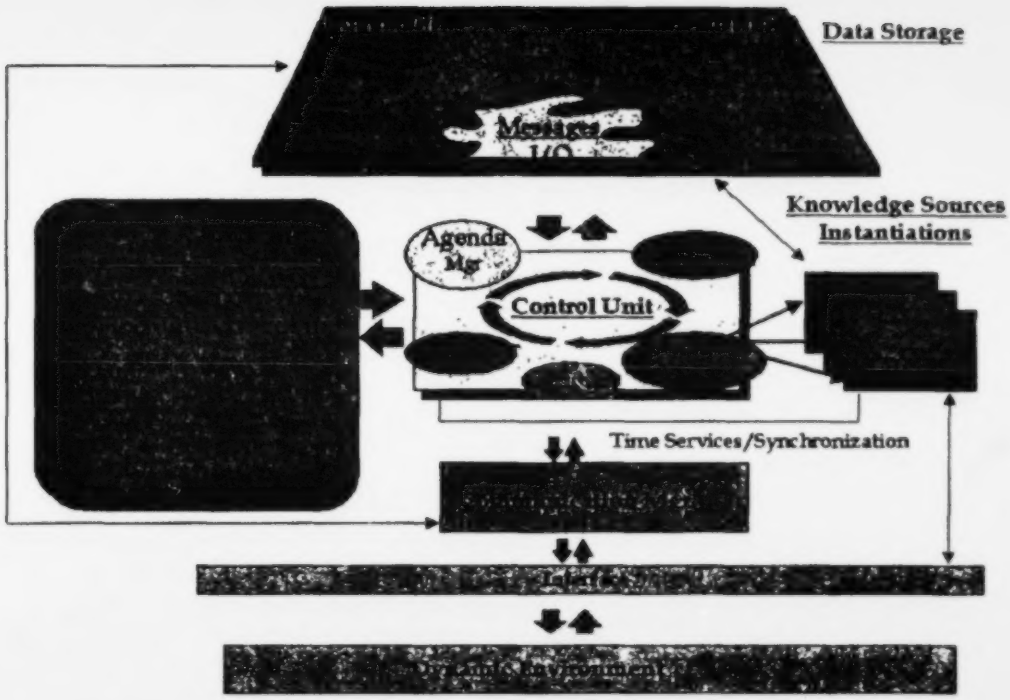

FIGURE 1 - Concurrent parallel blackboard architecture for an adaptive intelligent system

3.1 Blackboard data storage

The shared working memory or data storage (blackboard) through which the blackboard system supports information transactions is based upon an object-oriented representation. From reported work on object-oriented database management systems [13], [14] and distributed systems [15] and partly inspired from various schemes used for blackboard locking and data consistency reinforcement [16-18], a specific approach has been engineered to address concurrency control issues. Locking protocols and mechanisms (deadlock avoidance, data consistency and integrity), optimistic/pessimistic data access, recursive invocation (involving multiple object locking), and effects of class inheritance, object encapsulation and delegation on locking protocols were some of the

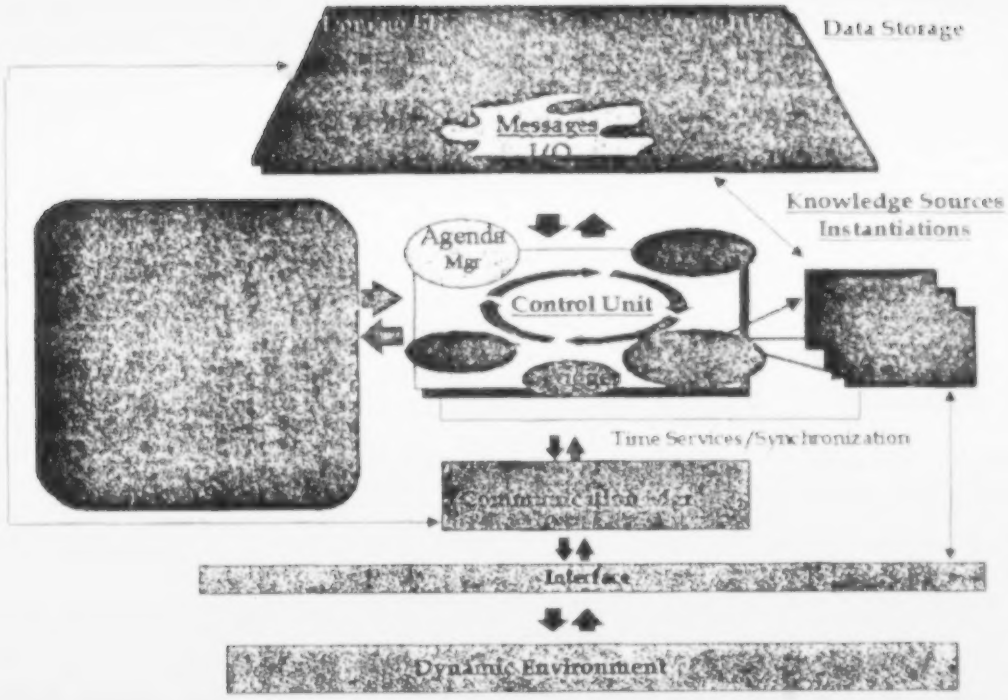

FIGURE 1 - Concurrent parallel blackboard architecture for an adaptive intelligent system

3.1 Blackboard data storage

The shared working memory or data storage (blackboard) through which the blackboard system supports information transactions is based upon an object-oriented representation. From reported work on object-oriented database management systems [13], [14] and distributed systems [15] and partly inspired from various schemes used for blackboard locking and data consistency reinforcement [16-18], a specific approach has been engineered to address concurrency control issues. Locking protocols and mechanisms (deadlock avoidance, data consistency and integrity), optimistic/pessimistic data access, recursive invocation (involving multiple object locking), and effects of class inheritance, object encapsulation and delegation on locking protocols were some of the

various issues that have been considered. A partitioned blackboard (data storage) architecture combined to the use of basic locking mechanisms are part of the proposed solution.

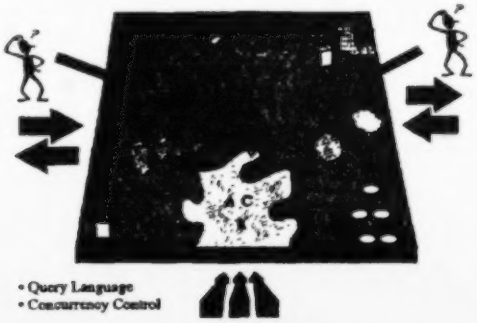

FIGURE 2 - Blackboard data storage

The blackboard data storage contains information that can be divided to various chunks of knowledge grouped into structured knowledge bases involving various abstraction levels (Figure 2). Information retrieval through complex requests formulated through a basic query language can be performed on searchable object characteristics such as types and attributes, organized in suitable data structures (B-tree and hash table). Concurrency is ensured through an autonomous locking strategy in which transactions acquire and release locks via operations on shared-memory (protocols). However, concurrency can be disabled to allow conventional locking to directly serialize the transactions and manage locking, similarly to a virtual blackboard data manager guaranteeing data consistency and deadlock-free transactions. Concurrent reading and exclusive writing transactions by knowledge sources are synchronously controlled by a locking mechanism (lock acquisition and release) encapsulated in a specialized object called monitor. Resorting to the use of low-level locking mechanisms (condition variables and mutex [19]), a monitor is a specific serialization mechanism preserving data integrity. It supports recursive object locking management and synchronization with a time delay (time-out). The monitor mediates also the event generation process when an

object is modified. As for the particular blackboard object access strategy chosen in handling complex requests (multiple object lock acquisitions), an optimistic scheme has been preferred (over the pessimistic one) given the expected cost associated with object contention for the class of problems to be investigated. Even though this specific choice increases the potential risk of deadlock situations, this problem can be alleviated by using a synchronization mechanism with time out.

3.2 Knowledge sources

Problem-solving domain/control knowledge or expertise required to achieve specialized tasks are embodied within knowledge sources. A knowledge source is defined through a description part involving an identifier, binding variables and data, a set of parameters and condition attributes as well as an action part. Specification characteristics include some kind of condition-action scheme in which the action is applicable, depending on the state of the blackboard (world). A triggering condition (or a goal purpose) refers to the potential interest or relevance for a knowledge source to trigger on a particular blackboard state change (data/objects) associated with an incoming blackboard event or goal. As a result, a knowledge source instantiation (KSI) data structure is created on the blackboard. A KSI denotes a pair referring to a knowledge source and a context (bound variables). A precondition specifies additional requirements that must prevail upon the domain blackboard to make the KSI previously generated applicable or executable. When both the triggering condition and the precondition are satisfied the KSI can then be selected for execution. However, as some execution delays are possible, obviation conditions are periodically examined as part of the control loop to eventually remove KSIs from the agenda should the blackboard state becomes incompatible with the expected activation conditions (previously satisfied precondition). The execution of a knowledge source corresponds to the activation of the action part aimed at changing the state of the blackboard (object creation, modification, deletion, goal posting, subgoaling). The action refers to the executable content of the knowledge source, a procedure comprising particular domain/control knowledge. As a contribution

to the problem-solving process, the action outcome, can then be shared by other applicable knowledge sources through asynchronous communication through the blackboard, creating new opportunities.

The proposed parallel blackboard-based adaptive intelligent system is characterized by the concurrent execution of knowledge sources. Concurrency is currently limited to inter-KS parallelism [9] as intra-knowledge source parallelism in which the action part involves parallel algorithms is being explored. Implemented through multithreading, a knowledge source embeds an exception handler mechanism to support interruptibility. The knowledge sources are currently implemented as procedural processing methods but can easily include rule-based reasoning methods. Ongoing efforts are directed toward the development of anytime [20] knowledge sources to address resource-bounded reasoning.

3.3 Control unit

The control unit is a mediating component between competing knowledge sources. Embedded as a separate control thread, the control unit involves the serial execution of the following actions or steps, namely, knowledge source triggering, goal management, agenda management, knowledge source scheduling and execution (interpreter). Control steps or procedures are generally achieved through specialized components (e.g. goal manager, agenda manager, meta-level controller or scheduler, interpreter) encapsulating all required functions to perform successfully and supervise/manage properly action execution. The basic control unit is pictured in Figure 3.

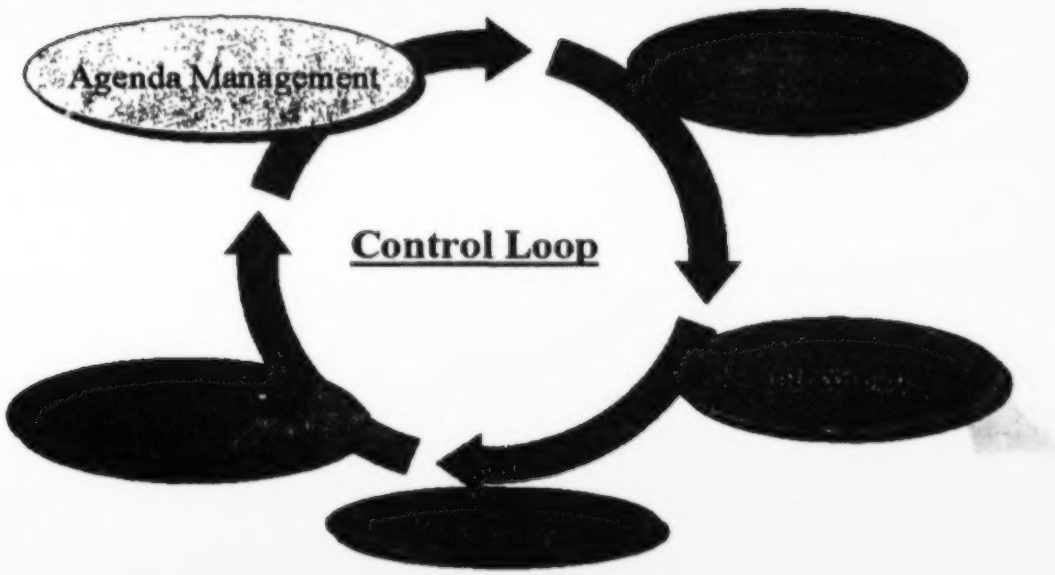

FIGURE 3 - Blackboard control unit (control cycle)

3.3.1 Triggering

As the first step of a control cycle, the triggering phase sequentially processes new blackboard input and verifies triggering conditions of all potentially relevant knowledge sources (Figure 3). A knowledge source triggers exclusively on events (event-driven) or on goals (goal-driven), depending, on whether, a triggering condition input is based upon a blackboard event (cognitive/perceptual) or a goal (domain/control). An event represents a state transition of the blackboard. Typically associated to a transaction on an object, an event refers to a change of information rather than the information per se. Triggering consists in extracting relevant objects from the event/goal input and performing condition verification (Figure 4). On satisfaction of triggering conditions a knowledge source is

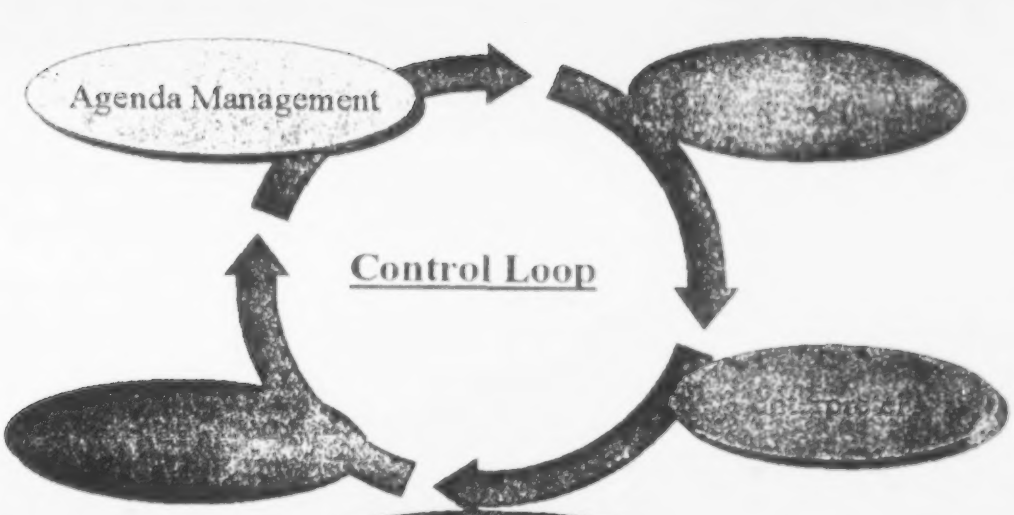

FIGURE 3 - Blackboard control unit (control cycle)

3.3.1 Triggering

As the first step of a control cycle, the triggering phase sequentially processes new blackboard input and verifies triggering conditions of all potentially relevant knowledge sources (Figure 3). A knowledge source triggers exclusively on events (event-driven) or on goals (goal-driven), depending, on whether, a triggering condition input is based upon a blackboard event (cognitive/perceptual) or a goal (domain/control). An event represents a state transition of the blackboard. Typically associated to a transaction on an object, an event refers to a change of information rather than the information per se. Triggering consists in extracting relevant objects from the event/goal input and performing condition verification (Figure 4). On satisfaction of triggering conditions a knowledge source is

instantiated. It is worth mentioning that multiple instantiations of the same knowledge source may also be possible for a particular input event.

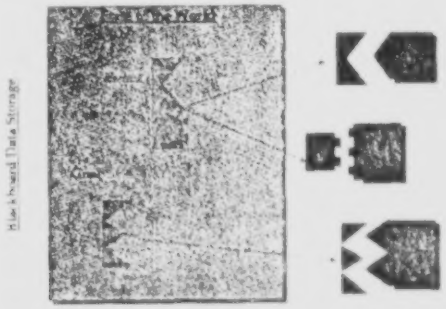

FIGURE 4 - Knowledge source triggering

The data structure of a knowledge source instantiation (KSI) conveys all relevant information such as the particular triggering event or goal, and the associated object bindings. Specific information has been added to enrich event description. Event representation has been extended from the original BB1 event concept to include user-defined type specification and a time-tag attribute to indicate the creation time of an event, information that may benefit some specialized knowledge sources.

On triggering phase termination, proper bookkeeping (event lists and data structures maintenance) is achieved, leading to the second phase of the control cycle, namely goal management.

3.3.2 Goal management

As an extension to event-driven control and an alternate mechanism for KSI generation (goal KS pairing), the concept of goal has been explicitly added to enrich blackboard control. Goal-directed control provides more flexibility for goal-driven problem-solving enhancing reactive event-driven control originally introduced to the blackboard framework. Control strategies/techniques based on goal-directed reasoning style such as planning-based control can then be designed more easily.

instantiated. It is worth mentioning that multiple instantiations of the same knowledge source may also be possible for a particular input event.

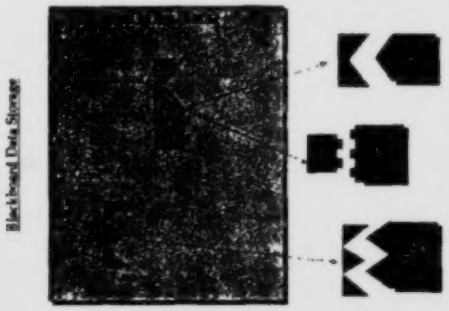

FIGURE 4 - Knowledge source triggering

The data structure of a knowledge source instantiation (KSI) conveys all relevant information such as the particular triggering event or goal, and the associated object bindings. Specific information has been added to enrich event description. Event representation has been extended from the original BB1 event concept to include user-defined type specification and a time-tag attribute to indicate the creation time of an event, information that may benefit some specialized knowledge sources.

On triggering phase termination, proper bookkeeping (event lists and data structures maintenance) is achieved, leading to the second phase of the control cycle, namely goal management.

3.3.2 Goal management

As an extension to event-driven control and an alternate mechanism for KSI generation (goal-KS pairing), the concept of goal has been explicitly added to enrich blackboard control. Goal-directed control provides more flexibility for goal-driven problem-solving enhancing reactive event-driven control originally introduced to the blackboard framework. Control strategies/techniques based on goal-directed reasoning style such as planning-based control can then be designed more easily.

A generic goal involves many attributes, including a description, an optional priority, a type, a status, a parent goal (supergoal) and a subgoal hierarchy to capture goal relationship as well as related knowledge sources (goal originators) if any, and a function or an operator to verify goal satisfaction. Goal types currently include the following: Achieve, Verify, Maintain and Try. These types include specific constraints imposed on goal achievement procedures. Hence, an "Achieve" goal type requires candidate KSs to be successively executed until the goal is reached. If all KSs fail, the goal is considered unreached and disregarded (failure). A "Verify" goal type focuses on the selection of one and only one KS candidate to determine the truth value of a goal description (predicate). A "Maintain" goal type is similar to "Achieve", except that on failure condition the goal is re-activated to allow new KS triggering. Finally, a "Try" goal type provides support for goal re-activation at any time, making possible new KS triggering, even though all the initial KS candidates have not been executed yet. This provides the opportunity for any KS triggering to be delayed should the situation subsequently become favorable (evolution of the world state). Goals are generated by domain knowledge sources and/or the meta-level controller. A knowledge source can also perform subgoaling as intermediate execution steps.

As part of goal management, the immediate step following the triggering phase is primarily concerned with goal achievement verification as new facts (blackboard world state) are compared with the blackboard. Goal satisfaction is typically related to a condition on the state of the blackboard world domain (e.g. a condition on a domain object). This particular task is performed by the goal manager, generally responsible for tracking/managing the evolution of a goal through the various states of its lifecycle from generation, to selection, and then, to goal achievement or failure (whether the associated action is already terminated or not). More specifically, goal verification is achieved by checking an attribute value of a blackboard domain object. When all subgoals of a higher level goal are reached, the latter is assumed to be achieved and the goal manager updates

its state accordingly making it inactive (Figure 5). Goal relationships on the goal structure are then modified and KSIs agenda status updated in order to maintain data consistency.

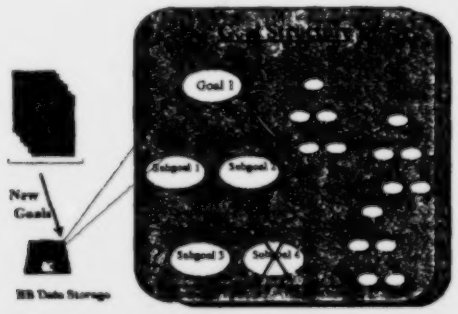

FIGURE 5 - Goal management

Further bookkeeping by the goal manager may also initiate a chain of events that could trigger specialized knowledge sources in updating the blackboard and any possibly connected external databases. Beyond the goal achievement verification step of the control loop, the goal manager also interacts with other control components, processing and transiting goals from one state to another (new, activated, achieved, failed) during different phases of control.

3.3.3 Agenda management

The agenda management phase consists in storing and transiting some KSI instances from one state to another as new incoming events and goals are generated (Figure 6). This task is carried out by the agenda manager. Generated KSIs on the agenda data structure can successively evolve through various states, namely, "triggered", "obviated" (no longer executable), "executable" when a KSI can be activated (pending execution), "scheduled", "executing", "suspended", "interrupted" and "executed" (action completion).

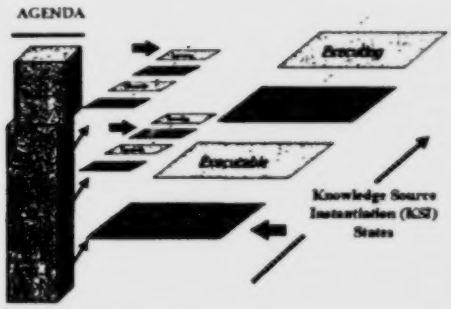

FIGURE 6 - Agenda management

Agenda management first consists in storing new generated KSI on the agenda into a "triggered" state. Then, the agenda manager checks the obviation conditions for all agenda KSIs, except for those already inactivated (interrupted, executed or obviated). If obviation conditions are satisfied the concerned KSIs are disregarded and moved to an "obviated" state. This process may be expensive and create a potential bottleneck for the control unit as the number of KSIs to be verified grows. This problem may easily be alleviated using various approaches such as process parallelization or by making KS obviation conditions event-selective. The other triggered KSIs that do not satisfy obviation conditions are moved to an "executable" state and therefore are eligible for further activation. The agenda can be further updated through the agenda manager by alternate components during other control phase. However, agenda management is primarily aimed at preparing the next control phase, the choice of KSI(s) to be scheduled for execution.

3.3.4 Scheduling (meta-level control)

The next phase of the control cycle has been primarily associated with action selection or knowledge source instantiations scheduling. Based upon control knowledge, the scheduler determines the next knowledge source(s) to be executed (or scheduled) while ensuring domain and computational resource constraints are met. Encompassing the classical features of the usual blackboard scheduler, our approach proposes to

its state accordingly making it inactive (Figure 5). Goal relationships on the goal structure are then modified and KSIs agenda status updated in order to maintain data consistency.

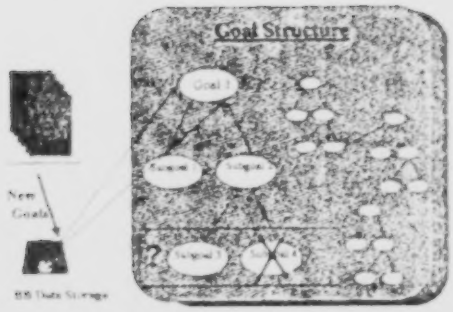

FIGURE 5 - Goal management

Further bookkeeping by the goal manager may also initiate a chain of events that could trigger specialized knowledge sources in updating the blackboard and any possibly connected external databases. Beyond the goal achievement verification step of the control loop, the goal manager also interacts with other control components, processing and transiting goals from one state to another (new, activated, achieved, failed) during different phases of control.

3.3.3 Agenda management

The agenda management phase consists in storing and transiting some KSI instances from one state to another as new incoming events and goals are generated (Figure 6). This task is carried out by the agenda manager. Generated KSIs on the agenda data structure can successively evolve through various states, namely, "triggered", "obviated" (no longer executable), "executable" when a KSI can be activated (pending execution), "scheduled", "executing", "suspended", "interrupted" and "executed" (action completion).

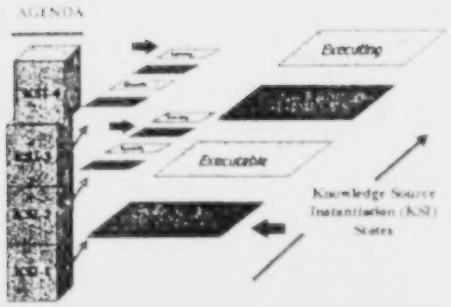

FIGURE 6 - Agenda management

Agenda management first consists in storing new generated KSI on the agenda into a "triggered" state. Then, the agenda manager checks the obviation conditions for all agenda KSIs, except for those already inactivated (interrupted, executed or obviated). If obviation conditions are satisfied the concerned KSIs are disregarded and moved to an "obviated" state. This process may be expensive and create a potential bottleneck for the control unit as the number of KSIs to be verified grows. This problem may easily be alleviated using various approaches such as process parallelization or by making KS obviation conditions event-selective. The other triggered KSIs that do not satisfy obviation conditions are moved to an "executable" state and therefore are eligible for further activation. The agenda can be further updated through the agenda manager by alternate components during other control phase. However, agenda management is primarily aimed at preparing the next control phase, the choice of KSI(s) to be scheduled for execution.

3.3.4 Scheduling (meta-level control)

The next phase of the control cycle has been primarily associated with action selection or knowledge source instantiations scheduling. Based upon control knowledge, the scheduler determines the next knowledge source(s) to be executed (or scheduled) while ensuring domain and computational resource constraints are met. Encompassing the classical features of the usual blackboard scheduler, our approach proposes to

extend/refine the classical scheduling scheme by introducing meta-reasoning in the KSI selection process and then allow explicit timely reasoning about actions (domain and control). This is aimed at adding intelligent behavior (situated reasoning [5], [21]) and resource-bounded reasoning capabilities to the control loop in order to support adaptive resource (computational and domain resources) planning and control within a time-varying uncertain environment. In that respect, a promising avenue to be further explored includes utility-driven control derived from a particular form of planning-based control (goal-directed control technique) involving anytime computation [20], in which a meta-level controller component performs deliberation scheduling and run-time monitoring on the next action(s) (knowledge source(s) instantiations) to be selected [22, 23]. Deliberation scheduling determines when, and how long (run-time allocation) a task (anytime knowledge source instantiation) should run (Figure 7). The value of computa-

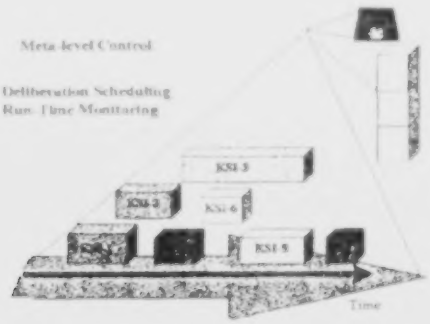

FIGURE 7 - Scheduling

tion determined through run-time monitoring relies on a variety of conditions and parameters, namely, the uncertainty regarding the quality of the results using knowledge source algorithm performance profiles, the state of the problem-solving process (quality of the current best computed plan so far), the current state of the environment (blackboard domain objects), the time pressure (change in the state of the environment), and the uncertainty regarding the environment (occurrence of stochastic events and use of approximate models). The meta-level controller component also achieves execution

extend/refine the classical scheduling scheme by introducing meta-reasoning in the KSI selection process and then allow explicit timely reasoning about actions (domain and control). This is aimed at adding intelligent behavior (situated reasoning [5], [21]) and resource-bounded reasoning capabilities to the control loop in order to support adaptive resource (computational and domain resources) planning and control within a time-varying uncertain environment. In that respect, a promising avenue to be further explored includes utility-driven control derived from a particular form of planning-based control (goal-directed control technique) involving anytime computation [20], in which a meta-level controller component performs deliberation scheduling and run-time monitoring on the next action(s) (knowledge source(s) instantiations) to be selected [22, 23]. Deliberation scheduling determines when, and how long (run-time allocation) a task (anytime knowledge source instantiation) should run (Figure 7). The value of computa-

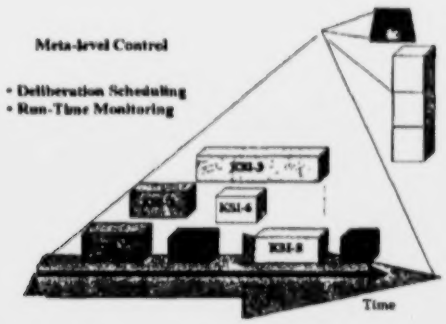

FIGURE 7 - Scheduling

tion determined through run-time monitoring relies on a variety of conditions and parameters, namely, the uncertainty regarding the quality of the results using knowledge source algorithm performance profiles, the state of the problem-solving process (quality of the current best computed plan so far), the current state of the environment (blackboard domain objects), the time pressure (change in the state of the environment), and the uncertainty regarding the environment (occurrence of stochastic events and use of approximate models). The meta-level controller component also achieves execution

P516658.PDF [Page: 28 of 44]

monitoring (monitoring planning through a meta-level control of computation, plan execution) while reasoning about the interleaving of plan execution and plan construction in order to guarantee response time. Execution monitoring involves feedback to compensate for the system's inability to accurately predict the effects of the control action on the behavior of a controlled process. Feedback relies on being able to accurately monitor the behavior of a process. Monitoring plan execution consists in recognizing information which satisfactorily supports or significantly violates expectations projected at run-time and in properly handling situations that tend to deviate from predicted ones (mismatch). The comparison between the real and the predicted situation is based on an approximate predictive model of the environment, feedback and feedforward information, belief state and background knowledge (probability, domain knowledge).

Incorporating the traditional event-driven approach (input-based-action selection), the current control unit implementation has been enhanced by adding explicitly goal-directed reasoning style using a simple form of planning-based control. This particular goal-directed control technique uses a meta-level control mechanism to support resource-bounded reasoning. As a first step, a primitive meta-level control model (planning-based control scheme) is being embodied as part of the scheduling component for simplification purposes.

3.3.5 Interpreter

The interpreter is mainly responsible for scheduling, resuming and interrupting the execution of a selected knowledge source (Figure 8). A knowledge source can be activated by specifying scheduling information and related execution parameters (e.g. start time, run-time) or characteristics. The interpreter also interacts with other control components to maintain blackboard data structures consistency when required (action completion, interruption). The current implementation of the interpreter involves a Task Scheduler object responsible for scheduling tasks (KSs) at the appropriate time. Designed to explicitly initiate serial and concurrent execution the Task Scheduler uses a separate

P516658.PDF [Page: 29 of 44]

thread to monitor the next task(s) to be activated and manage the task queue. The interpreter and its associated components capture specialized object classes and methods borrowed from the ACE (Adaptive Communication Environment) [19] library to ensure task scheduling execution/interruption as well as to modify information on a task queue when necessary or desired. In that respect, some of the capabilities offered by ACE in terms of flexibility in integrating task activation and interruptibility (e.g. managing pending locks when interrupting knowledge source execution), and the versatility of synchronization mechanisms, represent key elements to the interpreter implementation. ACE is a C++ toolkit encapsulating and enhancing the lightweight concurrency mechanisms provided both by Solaris 2.x threads [24] and POSIX Pthreads [25].

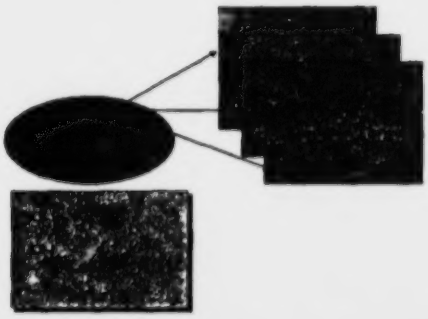

FIGURE 8 - Interpreter

3.4 Communication

Agent-to-agent communication is achieved through a communication manager. This component allows a blackboard agent to communicate with other agents (e.g. an alternate blackboard), external feeds or a database (Figure 9). The communication manager offers services (send) using low-level functions embedded in communication libraries to support agent interaction at the transport level. The implemented communication manager object is specified by a network name, the selection of the desired communication modes (the system currently supports IPC (interprocess communication) and KQML communication links) and an execution parameter indicating

whether specialized communication manager services are executed concurrently as separate threads or sequentially as part of the main control loop thread.

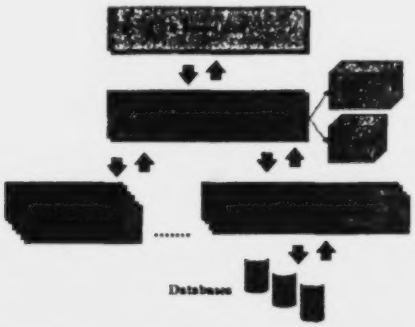

FIGURE 9 - Communication manager

The communication manager can manage synchronous and asynchronous communication through IPC and KQML links. The IPC link allows the blackboard system to connect to an IPC server and maintain an on-line feed through an active channel. External feed can then be maintained using socket functions. IPC does not require any specific message format. However, an IPC link imposes message routing or transactions to be mediated by the IPC server to ensure agent communication. The second communication mode supported by the communication manager allows a higher level message exchange between agents through KQML [26] language specifications. An agent name server (ANS) has been developed to ensure a direct communication between interacting agents. The agent name server or equivalently the KQML server is an intermediate component responsible for resolving agent names by broadcasting newly registered agent addresses (or network names) to other registered KQML agents. This capability then provides an easy way to support agent-to-agent communication while minimizing mediator interference and cost. Admissible transport protocols for KQML message routing include HTTP and TCP-IP. KQML stands for "knowledge query and manipulation language" [26], a language that supports interacting applications or agents to communicate "attitudes", in order to process transmitted information. Attitudes

thread to monitor the next task(s) to be activated and manage the task queue. The interpreter and its associated components capture specialized object classes and methods (borrowed from the ACE (Adaptive Communication Environment) [19] library to ensure task scheduling execution/interruption as well as to modify information on a task queue when necessary or desired. In that respect, some of the capabilities offered by ACE in terms of flexibility in integrating task activation and interruptibility (e.g. managing pending locks when interrupting knowledge source execution), and the versatility of synchronization mechanisms, represent key elements to the interpreter implementation. ACE is a C++ toolkit encapsulating and enhancing the lightweight concurrency mechanisms provided both by Solaris 2.x threads [24] and POSIX Pthreads [25].

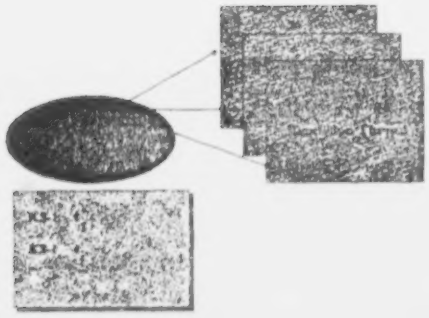

FIGURE 8 - Interpreter

3.4 Communication

Agent-to-agent communication is achieved through a communication manager. This component allows a blackboard agent to communicate with other agents (e.g. an alternate blackboard), external feeds or a database (Figure 9). The communication manager offers services (send) using low-level functions embedded in communication libraries to support agent interaction at the transport level. The implemented communication manager object is specified by a network name, the selection of the desired communication modes (the system currently supports IPC (interprocess communication) and KQML communication links) and an execution parameter indicating

whether specialized communication manager services are executed concurrently as separate threads or sequentially as part of the main control loop thread.

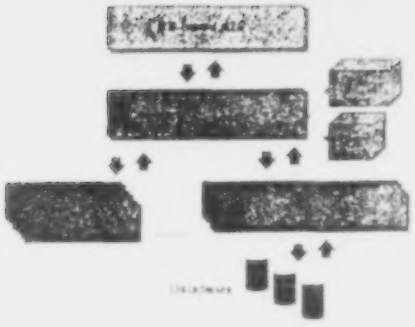

FIGURE 9 - Communication manager

The communication manager can manage synchronous and asynchronous communication through IPC and KQML links. The IPC link allows the blackboard system to connect to an IPC server and maintain an on-line feed through an active channel. External feed can then be maintained using socket functions. IPC does not require any specific message format. However, an IPC link imposes message routing or transactions to be mediated by the IPC server to ensure agent communication. The second communication mode supported by the communication manager allows a higher level message exchange between agents through KQML [26] language specifications. An agent name server (ANS) has been developed to ensure a direct communication between interacting agents. The agent name server or equivalently the KQML server is an intermediate component responsible for resolving agent names by broadcasting newly registered agent addresses (or network names) to other registered KQML agents. This capability then provides an easy way to support agent-to-agent communication while minimizing mediator interference and cost. Admissible transport protocols for KQML message routing include HTTP and TCP-IP. KQML stands for "knowledge query and manipulation language" [26], a language that supports interacting applications or agents to communicate "attitudes", in order to process transmitted information. Attitudes

P516658.PDF [Page: 31 of 44]

typically refer to proper abilities such as querying, requiring and achieving. KQML message structure involves a sender, a receiver, a performative (action to perform) and a content specification. A "performative" associated with a KQML message indicates an action to be performed by the receiver to properly process the information received from the sender. The content, unrestricted to a predetermined information format is represented through any language. KQML expressions can then involve subexpressions represented in a particular "content language". The KIF (knowledge interchange format) language is currently used to provide an object description format within the blackboard-based adaptive intelligent system. More information about KQML description and related features may be found in [26]. Inter-agent communication can therefore be initiated once proper connections and registration to the desired servers have been performed. The current communication manager implementation primarily benefits from basic services offered by the IPC (inter process communication) library provided by BB1 [12] as well as the KAPI (KQML application program interface) library [27].

The communication manager structure has a suitable interface component to translate data to be sent to or received from the environment. Embedded within the communication manager, the transaction manager is responsible for properly ensuring KQML message encoding and decoding and performing transactions. As a communication or message mediator/translator with the blackboard, its role is to encode or package information requests from the blackboard (e.g. through knowledge sources), decode incoming data from the environment in KQML and carry out transactions before modifying the blackboard (e.g. object creation/modification). Incoming messages to the blackboard requires an additional component to directly interact with the environment being typically represented through a database. Agent-to-database communication is achieved using a specialized environment data or database server before the messages transit through the transaction manager for further processing. Translated messages are then deposited on the blackboard. The current implementation involves a specialized data server with an ORACLE relational database to mediate transactions with the blackboard

P516658.PDF [Page: 32 of 44]

through the communication manager. It converts data using translation tables (object-database entity conversion) or restates KQML-KIF requests in useful representation to be interpreted using internal services. These services translate object information concerned by the request in a proper query language representation to support direct transactions with the database. Similarly, when monitored database information is modified, information is encoded into KQML messages and sent back to the blackboard through the transaction manager. The transaction manager is then responsible for decoding the message and taking the appropriate course of actions while depositing altered objects on the blackboard. It is worth mentioning, however, that contrarily to KQML messages, received IPC messages from the environment are directly deposited on the blackboard.

4.0 APPLICATION DOMAIN

The particular application to be investigated includes a class of tactical resource scheduling and vehicle routing/scheduling problems with time window constraints [28]. In the classical vehicle routing problem with time windows, a set of customers is served by a fleet of vehicles of limited capacity, initially located at a central depot. Each customer provides a period of time in which they require service, which may consist of repair work or loading/unloading the vehicle. The objective is to find tours for the vehicles, such that each customer is served in its time window, the total load on any vehicle is no more than the vehicle capacity, and the total distance traveled is as small as possible. The air vehicle tactical mission planning is a real-time generalization or extension of this problem in which an adaptive intelligent planner (resource controller) must plan, direct and control resources (tactical assets) in a time-varying uncertain and potentially unstructured (unpredictable) environment. The problem consists in planning incoming mission requests (transport, assistance, training) by routing air vehicles and scheduling the appropriate crew in order to satisfy specific criteria, subject to various set of constraints (temporal, operational, etc.). Criteria are related to measures of performance such as the rate of scheduled missions over a long time horizon (episodes), the number of scheduled missions over a certain time horizon (mission coverage) or the

number of requests in a minimum amount of time or throughput. Mission requests are assumed to be heterogeneous and generated randomly.

A closed-loop solution to a simplified version of the air vehicle tactical mission problem is being developed using the proposed blackboard-based AIS. It involves the contribution of various types of independent knowledge sources including planning KSs, monitoring KSs and dispatching KS (mission plan dissemination to the database). Planning KSs involve incremental planning and plan repair tasks. High-level planning KSs timely post a hierarchy of goals to be achieved through the execution of low-level planning KSs aimed at computing solutions. Hence, low-level planning KSs generate action plans to satisfy subgoals. Once an action has been planned/updated it is eventually disseminated (KS dispatcher) to the database and monitored. The monitoring task includes high and low-level KS monitors. The former monitors the world (domain) situation, whereas the latter performs execution monitoring, focusing its attention to a specific mission. KSs selection and/or scheduling is mediated by the control loop which determines which goal(s) to be pursued based upon a goal-directed control scheme. The system is interfaced to a tactical database to dynamically update desired information and set monitoring and control parameters.

5.0 RELATED WORK

Recent work in real-time artificial intelligence using blackboard technology is pervasive [4-6], [18], [29]. Among the most significant contributions related to the development of parallel blackboard system architectures focusing on hard real-time or mission-critical problems two are of particular interest to AIS. First, REAKT [29], a knowledge-based system built around a multi-agent, multi-task real-time architecture and then, the investigative efforts of Decker and Lesser's group on real-time control architectures. REAKT concepts mainly focus on concurrency control issues (blackboard data access) and real-time control strategies based upon progressive reasoning. The approach differs from our current AIS in many respects, including the use of an alternate

P516658.PDF [Page: 34 of 44]

dynamic scheduling mechanism, the absence of explicit goal representation, a unique event-driven knowledge source instantiation (KSI) mechanism, and its uniprocessing implementation. Part of the reported work by Decker and Lesser [18], [30] on real-time control architectures is primarily directed toward the use of approximate processing methods (KSs) and design-to-time scheduling [31] techniques and to achieve asynchronous control and coordination of scheduling activities through shared-memory parallel processing. Specifically, they investigated the effects of parallelism on blackboard system scheduling heuristics using different classes of control knowledge. Examined control knowledge included access collision (contention for blackboard access), KSI ordering (interdependence), KSI bottlenecking (opportunistic KSI execution favoring parallelism) and KSI invalidation (mutual exclusion of competing KSIs). Even though overall system performance improved, experiments conducted on parallel BB1-style control heuristics have not proved conclusive for all phases of problem-solving. We believe that the potential success of the approach is closely linked to the emergence of a suitable methodology to capture intricate goal relationship [32] between domain KSIs (interdependencies, resource constraint requirements inherent to their execution, interaction cost) taking into account a computable measure of intrinsic parallelism for application domains under consideration.

6.0 CONCLUSION

A blackboard-based approach for an adaptive intelligent system targeted to air vehicle tactical mission planning has been presented. An adaptive intelligent system is a system designed to modify its behavior in response to a dynamic and uncertain environment in order to successfully reach its goals. The proposed approach consists in evolving the opportunistic nature of the basic blackboard paradigm to support real-time dynamic planning and execution control in a time-varying and uncertain environment. The blackboard-based system mainly involves four major components, namely the blackboard (data storage), the knowledge sources, the control unit and the communication manager. The baseline architecture allows for the concurrent execution of multiple

P516658.PDF [Page: 35 of 44]

knowledge sources and provides explicit mechanisms to support event-driven and goal-directed (planning-based) problem-solving. As part of planning-based control, resource-bounded reasoning can be achieved through a meta-level control of computation responsible for deliberation scheduling and run-time monitoring of anytime knowledge sources. Based upon an object-oriented approach, a prototype of the baseline architecture has been partly implemented. It provides a suitable platform to initiate the investigation of real-time artificial intelligence concepts related to cognitive tasks such as planning and monitoring. This initiative is a significant contribution to the development of an automated real-time advisory decision support system for resource management.

Future work is directed to further refinements and extensions of the basic adaptive intelligent system design as well as performance analysis and impact study. Innovative ideas to be further developed and tested include (but not limited to) the integration of data and goal-directed control, the investigation of multi-level parallel control, the design of parallel knowledge source, cooperating knowledge sources through inter-control communication mechanisms and distributed knowledge source execution (problem-solving) in a distributed blackboard, application of artificial intelligence techniques and methods to handle uncertainty. This should pave the way to the exploration of advanced technology concepts leading to the elaboration of a suitable distributed blackboard framework.

7.0 ACKNOWLEDGMENTS

The author would like to thank very much Mrs. M. Bélanger and Mr. L. Lamontagne for their outstanding contribution, original ideas and enlightening discussions during the initial phase of the project. Both had a major impact in identifying key research areas and orienting future research directions. Their active involvement and suggestions significantly contribute to the quality of the reported work.

8.0 REFERENCES

1. [Nii, 1986] Nii, H.P., "The Blackboard Model of Problem Solving", AI Magazine, 7(2), pp. 38-53, 1986.

2. [Zilberstein, 1996] Zilberstein S., "Using Anytime Algorithms in Intelligent Systems", AI Magazine, 17(3), pp. 73-83, 1996.

3. [Musliner and al., 1995] Musliner D.J. and al., "The Challenges of Real-Time AI", IEEE Computer, January 1995.

4. [Garvey and Lesser, 1993a] Garvey A. and Lesser V., "A Survey of Research in Deliberative Real-Time Artificial Intelligence", TR 93-84, UMass Computer Science, November 1993.

5. [Velthuijsen, 1992] Velthuijsen H., "The Nature and Applicability of the Blackboard Architecture", PTT Research, 1992.

6. [Jagannathan and al., 1989] Jagannathan V. and al. (Eds),"Blackboard Architectures and Applications", Academic Press Inc., New York, 1989.

7. [Carver and Lesser, 1994] Carver N. and Lesser V., "Evolution of Blackboard Control Architectures", Expert Systems with Applications, 7, pp. 1-30, 1994.

8. [Englemore and al., 1988] Englemore R.S. and Morgan A.J. (Eds), "Blackboard Systems", New York, Addison-Wesley, 1988.

9. [Corkill, 1989] Corkill D.D., "Design Alternatives for Parallel and Distributed Blackboard Systems", in Jagannathan V. and al. (Eds),"Blackboard Architectures and Applications", Academic Press Inc., New York, 1989.

10. [Raulefs, 1989] Raulefs P., "Toward a Blackboard Architecture for Real-Time Interactions with Dynamic Systems", in Jagannathan V. and al. (Eds),"Blackboard Architectures and Applications", Academic Press Inc., New York, 1989.

11. [McManus and Bynum, 1996] McManus J.W. and Bynum W.L., "Design and Analysis Techniques for Concurrent Blackboard Systems", IEEE Trans. on Sys. Man and Cybern. - Part A, 26(6), pp. 669-680,1996.

12. [Hayes-Roth, 1985] Hayes-Roth B., "A Blackboard Architecture for Control", Artificial Intelligence, 26, pp. 251-321, 1985.

13. [Bertino and al., 1992] Bertino E. and al., "Object-Oriented Query Languages: The Notion and the Issues", IEEE Transactions on Knowledge and Data Engineering, 4(3), 1992.

14. [Molesky and al., 1994] Molesky L.D. and Ramamritham K., "Efficient Locking for Shared Memory Database Systems", Department of Computer Science, UMass, Amherst, TR-94-10, 1994.

15. [Cook, 1993] Cook D.J., "Strategies for Sharing Objects in Distributed Systems", Journal of Object-Oriented Programming, 1993.

16. [Fennell and Lesser, 1977] Fennell R.D. and Lesser V.R., "Parallelism in AI Problem-Solving: A Case Study of Hearsay-II", IEEE Transactions on Computers, 26(2), pp. 98-111, 1977.

17. [Ensor and Gabbe, 1985] Ensor J.R. and Gabbe J.D., "Transactional Blackboards", Proceedings of the Ninth International Joint Conference on Artificial Intelligence, pp. 340-344, 1985.

18. [Decker and al., 1993a] Decker K. and al., "A Real-Time Control Architecture for an Approximate Processing Blackboard System", International Journal of Pattern Recognition and Artificial Intelligence, 7(2), pp. 265-284, 1993.

19. [Schmidt, 1995] Schmidt D.C., "An OO Encapsulation of Lightweight OS Concurrency Mechanisms in the ACE Toolkit", WUCS-95-31, Washington University, St. Louis, 1995.

20. [Boddy and Dean, 1994] Boddy M. and Dean T., "Deliberation Scheduling for Problem Solving in Time-Constrained Environments", Artificial Intelligence, 67, pp. 245-285, 1994.

21. [Agre, 1990] Agre P.E., "Plans and Situated Actions: The Problem of Human-Machine Communications", Artificial Intelligence, 43, pp. 369-384, 1990.

22. [Zilberstein and Russell, 1993] Zilberstein S. and Russell S., "Anytime Sensing, Planning and Action: A Practical Model for Robot Control", International Joint Conference on Artificial Intelligence, 1993.

23. [Zilberstein, 1993] Zilberstein S., "Operational Rationality through Compilation of Anytime Algorithms", Dissertation, University of California at Berkeley, 1993.

24. [Eykholt and al., 1992] Eykholt J. and al., "Beyond Multiprocessing... Multithreading the SunOS Kernel", Proc. of the Summer USENIX Conference, San Antonio, Texas, June 1992.

25. [POSIX, 1992] IEEE, "Threads Extension for Portable Operating Systems (Draft 6), February 1992.

26. [Finin and al., 1995] Finin T. and al. "Specification of the KQML Agent-Communication Language", The DARPA Knowledge sharing initiative - External Interfaces Working Group. June 1995.

27. [EIT, 1996] Enterprise Integration Technologies Corporation and Lockheed Martin Missiles and Space Company, Inc. "KAPI", v2.6f, 1996.

28. [Solomon, 1987] Solomon M., "Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints", Operations Research, 35(2), 1987.

29. [Mensch, 1994] Mensch A., "REAKT: Real-Time Architecture for Time-Critical Knowledge-Based Systems", Intelligent Systems Engineering, pp. 153-167, Autumn 1994.

30. [Decker and al., 1993b] Decker K., Garvey A., Humphrey M. and Lesser V., "Control Heuristics for Scheduling in a Parallel Blackboard System," International Journal of Pattern Recognition and Artificial Intelligence, 7(2), 1993.

31. [Garvey and Lesser, 1993b] Garvey A. and Lesser V., "Design-to-time Real-Time Scheduling", IEEE Transactions on Systems, Man and Cybernetics, 23(6), 1993.

32. [Lesser and al., 1989] Lesser V. and al., "Goal Relationships and their Use in a Blackboard Architecture", in Jagannathan V. and al. (Eds),"Blackboard Architectures and Applications", Academic Press Inc, New York, 1989.

INTERNAL DISTRIBUTION

DREV TR-2000-157

1 - Director General

6 - Document Library

1 - Decision Aid Technologies Section Head

1 - Mr. J. Berger (author)

1 - Dr. A. Guitouni

1 - Mrs. M. Bélanger

1 - Dr. I. Abi-Zeid

1 - Dr. R. Breton

1 - Dr. A.-C. Bourry-Brisset

1 - Mr. V. Pille

1 - Mr. J. Roy

1 - LCol Josée-Ann Paradis

1 - LCdr D. Morrissey

1 - Major B. Deschênes

1 - Mr. D. Gouin

1 - Mr. R. Charpentier

1 - Mr. G. Thibault

1 - Mr. D. Demers

1 - Dr. G. Vezina

EXTERNAL DISTRIBUTION

DREV TR-2000-157

1 - DRDKIM

1 – DRDKIM (unbound copy)

1 - DRDC

1 - DSTA 3

Attn: Bob Elliott

1 - DAR 4

1 - DASP 4

1 - DAIM 4

1 - DLR 4

1 - DNR 4

1 - DCIEM

Attn: Ross Pigeau

8 - Air Command Headquarters

Westwin MB R3J 0T0

Attn: Charles Hunter

DCOS Ops Research

Attn: Dale Reding

CORA2

Attn: Dr Bao
CORA

Attn: LCol Hunter
A3 Plans

Attn: Major P.P.J. Lessard
G3 Plans

Attn: LCol Shawn E. Burtenshaw
2 Wing Operations Officer

Attn: Major C. Lynn Doucette
A3 Exercise Coordination 3

Attn: Major W.F. Seymour
A3 Exercice Coordination 2

1 - 3D Thrust Leader: Chris McMillan
DREO

2- NDHQ
Attn: Major T. Pascal, AFCCIS Project Manager;
Major D. Harper, AFCCIS Project

DOCUMENT CONTROL DATA

1. ORIGINATOR (name and address)	2. SECURITY CLASSIFICATION (Including special warning terms if applicable)
Jean Berger Defence Research Establishment Valcartier Decision Support Technology Section 2459 Pie-XI Blvd. North	Unclassified

3. TITLE (Its classification should be indicated by the appropriate abbreviation (S, C, R or U)
AN ADAPTIVE INTELLIGENT SYSTEM ARCHITECTURE FOR TACTICAL MISSION PLANNING

4. AUTHORS (Last name, first name, middle initial. If military, show rank, e.g. Doe, Maj. John E.)
Berger Jean

5. DATE OF PUBLICATION (month and year)	6a. NO. OF PAGES	6b .NO. OF REFERENCES
July 2000	39	32

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. Give the inclusive dates when a specific reporting period is covered.)
Tech Report

8. SPONSORING ACTIVITY (name and address)

9a. PROJECT OR GRANT NO. (Please specify whether project or grant)	9b. CONTRACT NO.
Project 3dm	

10a. ORIGINATOR'S DOCUMENT NUMBER	10b. OTHER DOCUMENT NOS
	N/A

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

- [x] Unlimited distribution
- [ ] Restricted to contractors in approved countries (specify)
- [ ] Restricted to Canadian contractors (with need-to-know)
- [ ] Restricted to Government (with need-to-know)
- [ ] Restricted to Defense departments
- [ ] Others

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected.)

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

Current tactical mission planning systems operating in a highly dynamic and uncertain environment can work in real-time but cannot guarantee efficient solutions. Potential constraints and conditions associated with mission criticality impose the development of adaptive reasoning capabilities to support resource planning, monitoring and control. A blackboard-based approach for an adaptive intelligent system targeted to air vehicle tactical mission planning is presented. The proposed approach consists in evolving the opportunistic nature of the basic blackboard paradigm to support real-time dynamic planning and execution control in a time-varying and uncertain environment. The parallel blackboard architecture allows for the concurrent execution of multiple knowledge sources and provides explicit mechanisms to support event-driven and goal-directed problem-solving. It also includes resource-bounded reasoning capabilities represented through a meta-level control of computation responsible for deliberation scheduling and run-time monitoring of anytime knowledge sources. Based upon an object-oriented approach, a prototype of the baseline architecture has been partly implemented. This initiative is a significant contribution toward the development of an automated real-time advisory decision support system for resource management.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

adaptive intelligent system, blackboard, real-time, dynamic planning, execution control, mission planning

#516658

CA012062